Neural Comput & Applic (2002)11:45–52

Neural Computing & Applications

Neural Networks for Autonomous Path-Following with an Omnidirectional Image Sensor

A. Rizzi[1], R. Cassinis[2] and N. Serana

[1]Department of Information Technology, University of Milano, Crema (CR), Italy; [2]Department of Electronics for Automation, University of Brescia, Brescia, Italy

This paper presents a path-following system implemented with two different types of neural networks, that enables an autonomous mobile robot to return along a previously learned path in a dynamic environment. The path-following is based on data provided by an omnidirectional conical visual system, derived from the COPIS sensor, but with different optical reflective properties. The system uses optical and software processing and a neural network to learn the path, described as a sequence of selected points. In the navigation phase it drives the robot along this learned path. Interesting results have been achieved using low cost equipment. Test and results are presented.

Keywords: Neural networks; Omnidirectional vision; Robotic navigation

1. Introduction

To perform an exploration in an unknown environment, it is safer for a robot to be able to return along the same exploratory path from which it originally travelled. In the meantime, the robot can collect useful information to support the exploration task. Different sensors, such as lasers, optical range finders, sonar or proximity sensors, can be used to scan the environment and obtain useful data in order to allow the robot to self-localise its position. Scanning a wide area around the robot with a vision system can be done mainly using two different approaches, a pan-tilt camera or an omnidirectional reflecting surface, whose shape and reflectance properties can differ widely [1–6], in this way affecting the visual information acquired. In fact, the surface of the omnidirectional device can either be a perfect mirror that reproduces the visual information around it, or a glazed surface, acting as a low pass filter on the image frequency components.

In an omnidirectional image, the advantage of capturing a wide area in a single snapshot is counterbalanced by the geometrical distortion of the omnidirectional projection. For this reason, in the proposed system, the omnidirectional vision is not used to recognise an object, thus making a perfect reflecting surface and a shape that corrects the projected geometrical distortions unnecessary and superfluous for the simplicity of the system.

Following this assumption, a different omnidirectional vision device called an Omnidirectional Pseudo Mirror (OPM) has been devised. The information coming from this sensor is stored in a learning system so that, once the path is learned, the robot can return along the very same path on which it originally travelled.

Section 2 presents an overview of the whole system. Section 3 describes the OPM in detail, Section 4 describes the preprocessing of the omnidirectional images, and Sections 5 and 6 explain the use of the learning system and its characteristics. Sections 7 and 8 report indoor tests results.

The proposed system derives from *Pollicino*, a previous self-localisation system [7,8] with a similar structure but with a different approach. *Pollicino* was designed to learn an area and localise itself inside. Its purpose was not to navigate, but only to self-localise in the whole learned area, while the

Correspondence and offprint requests to: Mr A. Rizzi, Department of Information Technology, University of Milano, Via Bramante 65, 26013 Crema (CR), Italy. Email: rizzi@dti.unimi.it

proposed system is designed to learn a single path (not an area) in a more compact way and drive the robot back along the same path. Both systems use the same OPM, but the learning systems and the use of images is different.

2. System Description

There are four subsystems within the system structure (see Fig. 1):

- the omnidirectional vision subsystem, composed of an omnidirectional image sensor (OPM) and a camera, which collects images 360° around the robot (Fig. 1, top left);
- the image preprocessing subsystem which simplifies the omnidirectional visual information images from the surrounding environment (Fig. 1, bottom left);
- a neural network which classifies the extracted data (Fig. 1, top right);
- an orientation correction subsystem which corrects the robot's orientation along the path (Fig. 1, bottom right).

The omnidirectional vision subsystem uses a colour CCD camera in axis with the OPM, which makes a first optical preprocessing with its glazed surface. From the grabbed image, a 360° array of RGB colour values is extracted, one for each degree of the omnidirectional vision field, to form a Horizon Vector (HV), which is used to train the neural network and estimate the robot's position along the path.

The Artificial Neural Network (ANN), fed with HVs, is capable of recognising one visual vector from another, even if noise is superimposed or if the HV is partially modified by obstacles or other moving objects. This property is very important if the system has to act in a dynamic environment.

Taking advantage of the ANN coding capability, the orientation control system compares the input HV with the one it learned, identifying precisely the image pattern shift.

Once the path is learned as a sequence of HVs, the robot can retrace its path simply, moving itself point-by-point following the sequence of the previously visited points and preserving the correct orientation.

This system is economical, using a low cost conical mirror with a non-perfect surface, and a system that does not require a high quality camera because of the OPM optical pre-processing mechanism. The algorithms are also very simple, and neural network training happens only once at the end of the navigational learning phase.

3. OPM, the Image Sensor

The Omnidirectional Pseudo Mirror is the same device as that used in the Pollicino system [8]. It is neither a perfect conical mirror nor it is designed to obtain a high definition image of the environment. The large amount of image detail with a perfectly reflecting mirror would increase the image's elaboration complexity. As the OPM surface increases its blur effect (see Fig. 2), the high frequency content in the image decreases [9]. The chosen OPM is the bottom right one of Fig. 2. The blurring effect and the mirror shape make the vertical dimension collapse along each ray of the image [10].

A reference system transformation from polar to rectangular is applied to this image to simplify the processing. Pixels around the centre of the image are discarded, avoiding aliasing problems due to the presence of a great number of sectors in a relatively

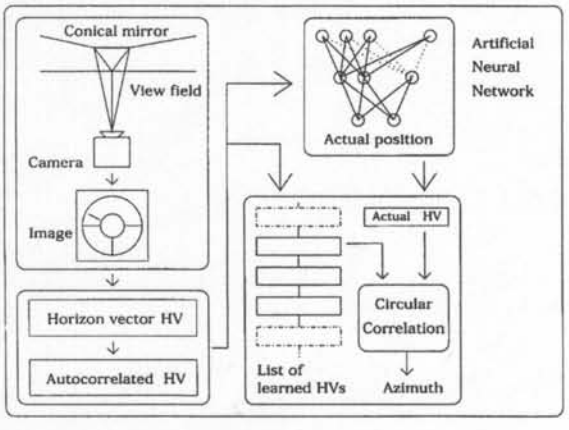


Fig. 1. The system structure.

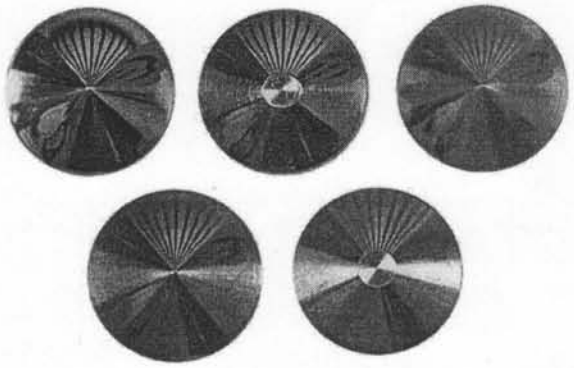


Fig. 2. Different conical mirrors surfaces.

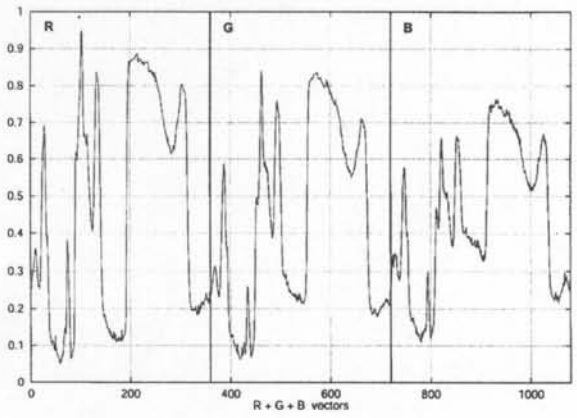


Fig. 3. An RGB *Horizon Vector* (HV).

small area. Then a selected ring of the circular image is mapped into a rectangle. Each vertical stripe of the rectangle contains all the pixels representing the chromatic characteristics perceived in the related direction. Each stripe is then averaged to obtain a single value for each degree of the view field (this is an HV). An example of HV displayed with the three RGB components one after the other is shown in Fig. 3.

A spectral analysis of the omnidirectional sensor output has been made to distinguish useful frequencies after pre-processing. Snapshots from different indoor environments have been taken, and the relative HVs have been extracted: the averaged periodogram of Fig. 4 shows that only normalised frequencies under 0.2 are significant.

In fact normalised frequencies over 0.05 are more than 40 dB under the continuous value. In other words, the cone reflectance acts as a low pass filter, allowing the sector decimation described in the next section.

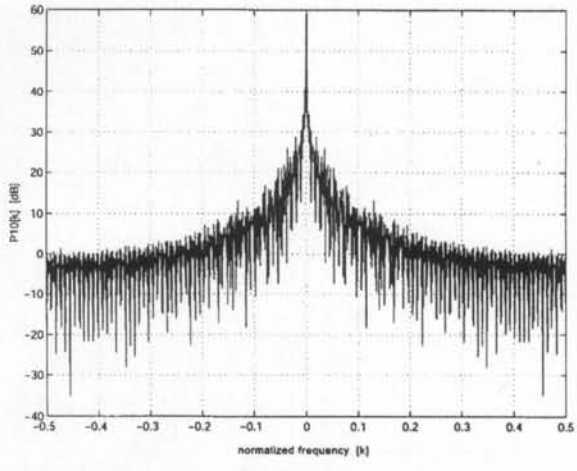


Fig. 4. The averaged periodogram of 12 HV from different environments.

4. The HV Preprocessing

Each HV is composed of RGB values. Thus, there are three resulting vectors, with 360 values each. For the neural network, an input layer of 1080 values is very large, and requires a long period of training. According to the spectral response of the image sensor (Fig. 4), a great reduction of the input layer can be made, thus we decided to undersample the HVs by a factor of 10. In fact, an indoor environment is typically distinguished by fixed objects, such as furniture, walls and doors, with a great amount of low frequency components. Thus, the vectors are low-pass filtered with a moving average filter, and the resulting vectors have 108 values, 36 for each chromatic component.

In the next step, each channel vector is circularly correlated with itself, and the continuous component is subtracted and finally normalised. An example is visible in Fig. 5, where the circular auto-correlation of the HV in Fig. 3 is shown. The choice of the circular correlation follows the geometric characteristics of the omnidirectional vector. The mean value is subtracted to keep only the 'shape' of the correlation. In this way, small brightness shifts can be tolerated, and the ANN can recognise a location even with illumination changes.

5. The Neural Networks

Two different neural networks with two different path point coding approaches have been developed to learn the HVs along the path. The first one,

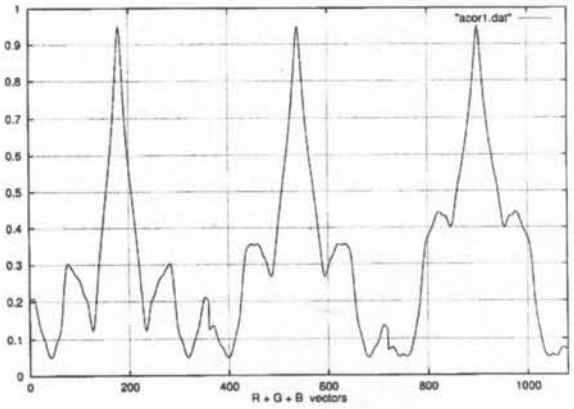


Fig. 5. The circular autocorrelation of the HV of Fig. 3.

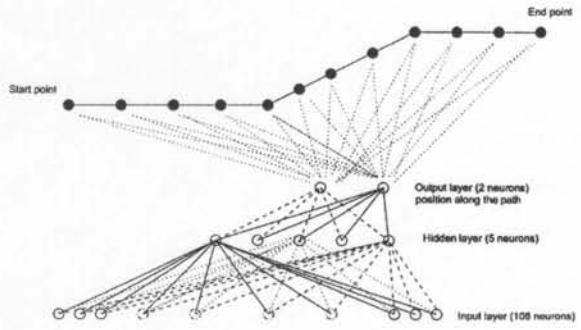


Fig. 6. The *NetA* structure.

called *NetA*, codes the sampled points along the path with different values on two output units (Fig. 6); the second one, called *NetB*, codes the path using one output unit for each sampled point along the path (Fig. 7). Both are feed-forward totally connected ANNs; for both, the activation function is sigmoidal, and they are trained with a standard back-propagation algorithm.

NetA (Fig. 6) has 108 input, 5 hidden and 2 output neurons. Using NetA to learn about 40 points in a path, only the first output neuron is used, while the other is kept unused. The number of points coded by a single output unit can be further investigated, but according to some preliminary tests, using both the output units, more than 1000 positions can be encoded. That should be enough even for long path sections in complex environments: with a 20 cm sampling step, more than 200 meters can be learned.

The structure of NetB (Fig. 7) changes according to the number of the points to learn. NetB dynamically changes the number of links and connections according to the characteristics of the environment. It has 108 input neurons and, with N sampled points along the path, $N/3$ hidden and N output neurons. A typical number of sampled points ranges from 30–70.

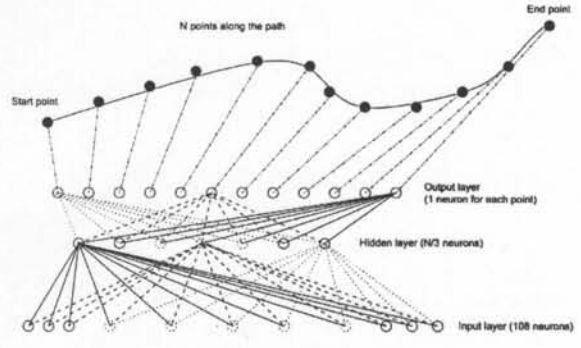


Fig. 7. The *NetB* structure.

Both the ANNs have been implemented in the C language to obtain good speed performances. Computational times are presented in Section 8. The amount of memory needed to store the data is very small. Each HV requires 108 bytes, since it consists of 36 24-bit samples. The amount of memory required to store all the HVs to learn a path depends upon the chosen sampling step. For a typical navigation task, 2–5 kbytes can be a reasonable estimate. The program that realises the neural network needs a floating point variable for each node and link. The number of bits for each variable affects the precision of the system. NetA has 108 inputs neurons, 5 hidden and 2 output; using a long double (12 byte using *gcc* on Pentium under Linux), the total amount of memory is less than 1 Kb. NetB has a changing number of nodes, but almost the same structure, thus the memory required is usually under 1 Kb.

6. The Orientation Correction Subsystem

The circular convolution of the HVs is used to estimate the robot's unwanted orientation changes. During the navigational training, all the HVs with their positions along the path are stored in an ordered list. When the system is used to follow the learned path, the neural network returns an estimate of the robot position along the path. This value usually falls in the middle of two sampled points. Given two vectors, the maximum correlation point indicates how much one vector is a shifted copy of the other. This property is used to estimate the robot shift. The correlation of the actual HV is compared both to the preceding and the following one in the list. A linear interpolation of the shift between the two circular correlation maximum points is used to estimate the actual shift with respect to the learned position. Using this estimate, a robot rotation is performed.

This estimate is robust toward limited changes in the environment. In fact, the correlation is not affected by noise that modifies only a part of the image (i.e. partial occlusions, moving objects, etc.). Moreover, moderate brightness changes are well tolerated due to the fact that the HV's shape is more significant than its absolute values (see Section 4).

7. Route Definition and Test Setup

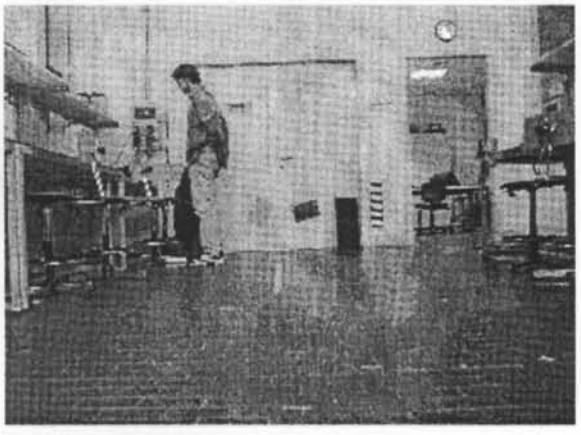
The system is designed to learn a route as a sequence of straight paths, each rotated from

5°–70° with respect to the previous one. The 5° lower bound has been chosen to stay up from the precision limit of the robot on which the system has been tested (Pioneer I by Activ Media Inc.); the upper limit is set not to reach the 90° turning back limit.

A path sampling step can vary from 10–50 cm from each point. If necessary, this sampling distance can be further varied if the environment characteristics originate HVs with a high variance. Different paths have been chosen inside the lab. Pictures of the test environments are visible in Fig. 8.

Using NetA, for each learned point along the path, the HV and its autocorrelation vector is extracted and labelled with a progressive floating point number into range [0–1]. Then the neural network is trained using the autocorrelation vectors as inputs and the label number as outputs. The training continues until a mean error of 1/200 is reached.

(*a*)



(*b*)

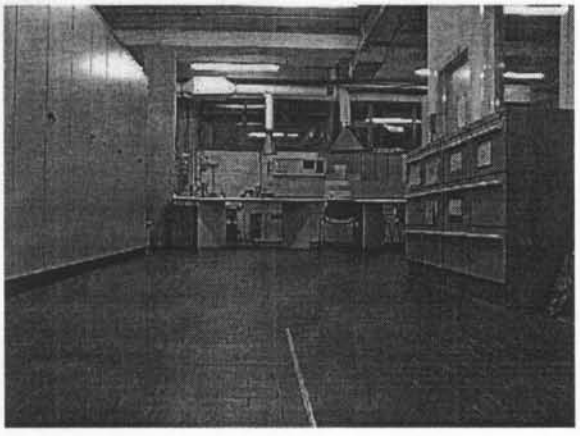


Fig. 8. Typical test environments.

NetB has an output neuron for each sampled point along the path, so the same input vectors are used to train the net with an active value only on the relative output neuron.

The navigation through the learned paths is performed after the training phase. The robot is positioned at the starting point, then a simple control system guides the robot towards the ending point of the path.

While the ANN estimates the actual robot position, the orientation control subsystem returns the azimuth difference. The azimuth needed corrections are stored into an accumulator that changes the robot heading only when a 5° value, or greater is reached. This prevents the system from zigzagging under the action of very small azimuth difference.

8. Tests and Results

The system has been mounted on a Pioneer I robot by Activ Media, carrying a Pentium 133 processor on its back, as shown in Figs 9 and 10.

The tests have been performed on 3–4 m long paths. This value can be extended to an arbitrary path length collecting a series of sub-paths with the relative ANN's weights (see Section 5).

Tests on different paths and on various orientation detections have been performed and the results are reported.

8.1. Straight Line Navigation

In these experiments, the performance of the system vs. the chosen sampling step has been tested. For this purpose, different straight line paths have been

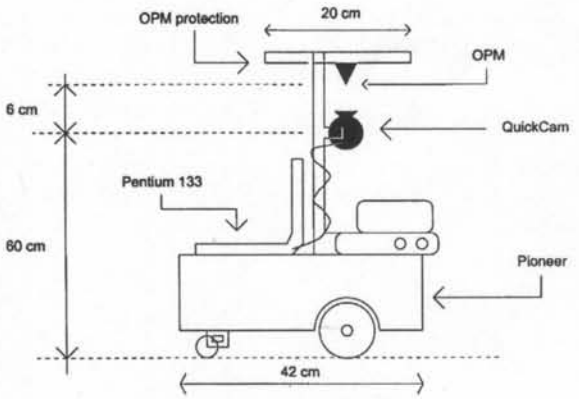


Fig. 9. Scheme of the robot with the omnidirectional vision system.

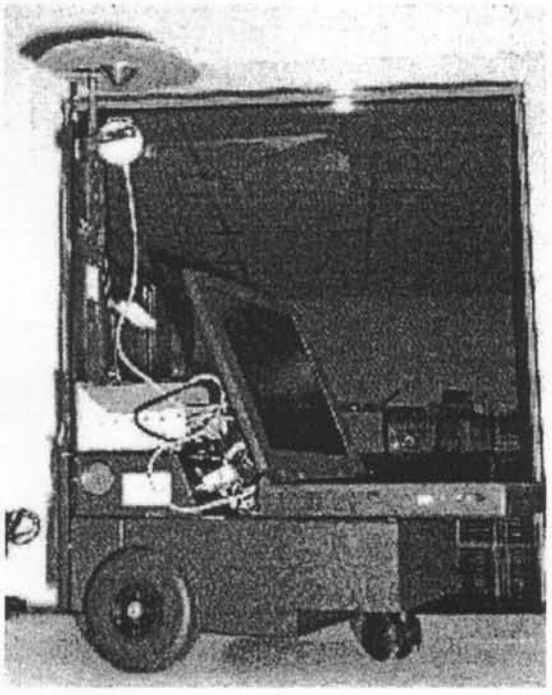


Fig. 10. The actual robot with the system mounted.

learned with different sampling steps, using both neural networks.

As shown in Figs 11–14, the sampling step in the learning phase moderately affects the system performance following a straight line. The two neural networks have shown almost the same performance, as can be noticed in the values of Table 1. In all the tested cases, the system follows the line with limited errors (see Figs 11–14), mostly due to the robot motor driving tolerance.

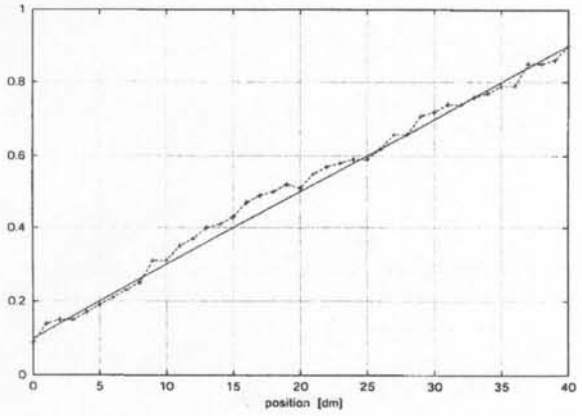


Fig. 11. Linear navigation with 20 cm sampling step using NetA.

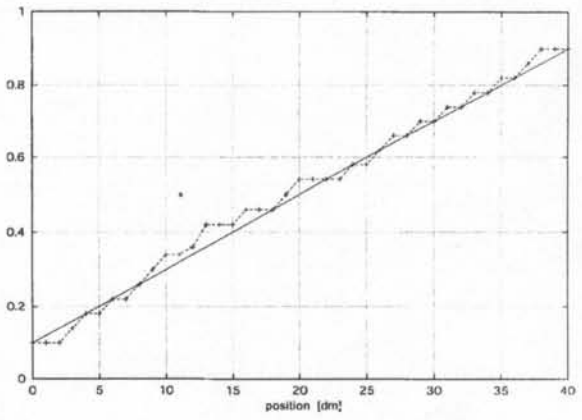


Fig. 12. Linear navigation with 20 cm sampling step using NetB.

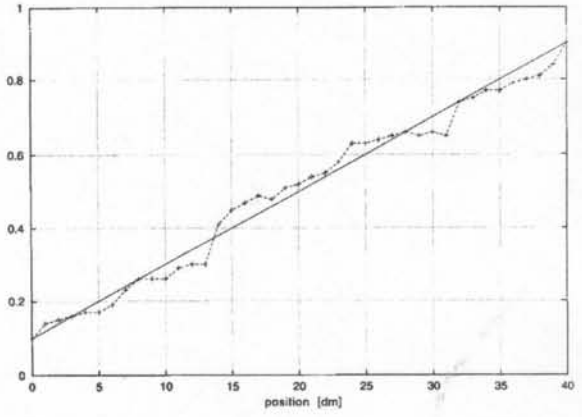


Fig. 13. Linear navigation with 40 cm sampling step using NetA.

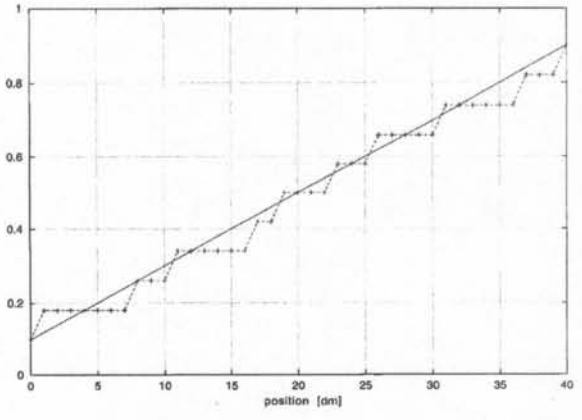


Fig. 14. Linear navigation with 40 cm sampling step using NetB.

8.2. Orientation

The orientation correction capabilities have been tested on learned points along the path (Table 2),

Table 1. Straight path results vs. sampling step.

	Step	Mean error	Regression	Max error
NetA	20 cm	0.52	0.995	1.25
	40 cm	0.35	0.981	1.25
NetB	20 cm	0.36	0.997	1.00
	40 cm	0.29	0.989	1.00

Table 2. Orientation estimate in learned points along the path.

Real orientation	Estimate orientation
0	0
20	18
45	39
70	66
90	89
130	125
180	170

Table 3. Orientation estimate in non-learned points along the path.

Real orientation	Estimate orientation
0	0
20	16
45	45
70	66
90	94
130	135
180	170

Table 4. Orientation estimate in points 50 cm distant from the path.

Real orientation	Estimate orientation
0	5
20	20
45	50
70	70
90	81
130	120
180	165

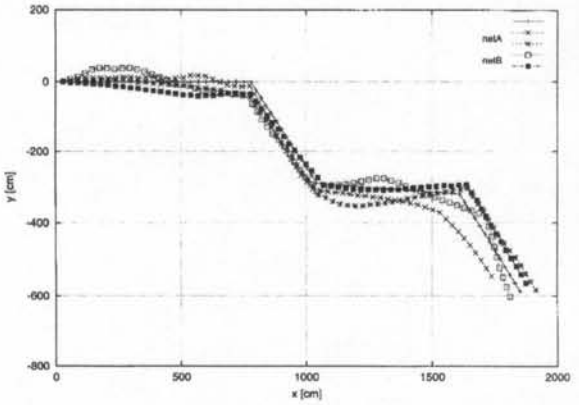


Fig. 15. Test navigations along a turning path.

on non-learned points along the path (Table 3), and on points 50 cm distant from the path (Table 4).

Estimate errors greater than 5° are written in bold. As can be noticed, in most cases, the orientation estimate error is below the robot turning limit (5°).

8.3. Navigations

The system has been tested on typical paths in the indoor environments shown in Fig. 8, in normal and noisy conditions (people moving around the robot, secondary light being switched on and off).

In Figs 15 and 16, two test navigations for each neural network, in two different environments, are presented. The continuous line indicates the learned paths, the lines with crosses represent the navigations with NetA, and the lines with squares represent the navigations with NetB. Note that the navigations have a similar performance.

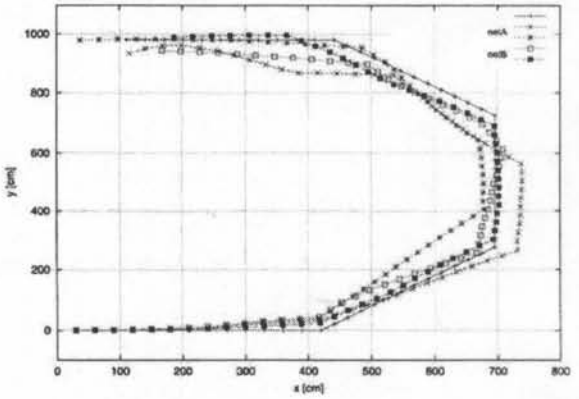


Fig. 16. Test navigations along a U-turning path.

Table 5. Mean error values for navigations in different conditions.

	NetA	NetB
normal	0.5	2.2
60 cm from origin	3.1	3.5
light changes	1.1	2.3
noise	1.4	3.6

Table 6. Computational times (in seconds).

Image processing	0.3
ANN path learning	120
ANN path estimate	0.0002
Orientation estimate	0.05

Table 5 presents the mean error values of different navigations in the following conditions: normal condition, with a starting point 60 cm from the origin of the learned path, with great changes in the lighting condition (lights switched on or off, window curtains drawn or not, etc.), and with some people walking around the robot during its navigation. The error is measured as a percentage of the sampling step, in this case 20 cm. As for the straight navigations, the mean values show little difference between the ANNs, showing that both are able to follow the learned path.

The computational times on a Pentium 133 processor, under Linux OS, can be seen in Table 6 (in seconds). The image processing time does not consider the image grabbing time. The path learning time has been measured on a 100 point path, and does not include the HV collection along the path.

9. Conclusion and Perspectives

This paper describes a path-following system that uses an omnidirectional image sensor to grab visual information from the environment and an artificial neural network to learn this information along the path. The visual information is preprocessed and compacted in monodimensional sequences called *Horizon Vectors* (HV), and the path is coded and learned as a sequence of HVs. After the learning phase, the system can guide the robot back along the path using the neural network position estimate and its orientation correction capability, performed by a simple circular correlation on the HVs.

A low cost conical Omnidirectional Pseudo Mirror (OPM) greatly reduces the visual input data, while the simple processing can be realised with low cost hardware.

The system can localise itself with good precision along a learned path, even in noisy conditions: its performance has been tested in indoor navigation, showing a good capability of following pre-learned paths, even with natural small changes around the robot (people walking, etc.).

References

1. Chahl JS, Srinivasan MV (1997) Range estimation with a panoramic visual sensor. J Opt Soc Am A 14
2. Franz MO et al (1998) Learning view graphs for robot navigation. Autonomous Robots 5: 111–125
3. Waxman AM, LeMoigne JJ, Srinvasan B (1987) A visual navigation system for autonomous land vehicles. IEEE J Robotics Automat 3(2): 124–141
4. Yagi Y, Nishizawa Y, Yachida M (1995) Map based navigation for a mobile robot with omnidirectional image sensor COPIS. IEEE Trans Robotic & Automat 11
5. Yagi Y, Fujimura S, Yachida M (1998) Route representation for mobile robot navigation by omnidirectional route panorama Fourier transformation. Proc IEEE International Conference on Robotics & Automation, Leuven, Belgium
6. Lin L, Hancock T, Judd JS (1998) A robust landmark-based system for vehicle location using low-bandwidth vision. Robotics and Autonomous System 25: 19–32
7. Cassinis R, Grana D, Rizzi A (2001) A perception system for mobile robot localisation. Machine Learning and Perception, vol 23. World Scientific, 1996
8. Rizzi A, Cassinis R A robot self-localisation system based on omnidirectional color images. *Robotics and Autonomous Systems* 34: 23–38
9. Nayar SK, Ikeuchi K, Kanade T (1991) Surface reflection: physical and geometrical perspectives. IEEE Trans Pattern Analysis Machine Intell 13(7)
10. Spagnolo FA, Brand KW (1968) Lambert scattering from a cone and paraboloid of revolution. *Applied Optics* 7(1)